Getting started with meta-selinux - enhancing system security on QEMU

Yocto Summit 2021.11

Tomasz Żyjewski

3MDEB

Agenda

- `whoami`
- Who we are?
- SELinux description
- Access Control Mechanisms
- Policies
- meta-selinux overview
- Image building
- Running in QEMU
- Working with access denials
- Summary
- Q&A

whoami

Tomasz Żyjewski
Embedded Systems Engineer

- @tomzy_0
- tomasz.zyjewski@3mdeb.com

- over 2 years in 3mdeb
- integration of update systems and OS creation for embedded devices
- interested in:
  - Yocto Project
  - OS updates
  - boot-time optimization

Who we are?

- coreboot licensed service providers since 2016
- coreboot project leadership participants
- UEFI Adopters since 2018
- Official consultants for Linux Foundation fwupd/LVFS project
- Yocto Participants and Embedded Linux experts
- Open Source Firmware enthusiasts and evangelists

SELinux description

- Security architecture for Linux systems that allows administrators to have more control over who can access the system, name comes from Security-Enhanced Linux (SELinux)
  - originally developed by the United States National Security Agency (NSA) as a series of patches to the Linux kernel using Linux Security Modules (LSM)
- Released to the open source community in 2000
  - integrated into the upstream Linux in 2003
  - last stable release (v3.3) comes in Oct 22, 2021 - can be found on github page https://github.com/SELinuxProject/selinux/releases
- Security-Enhanced Linux implements the Flux Advanced Security Kernel (FLASK)
  - kernel with architectural components prototyped in the Fluke operating system
  - provide general support for enforcing many kinds of mandatory access control policies

SELinux description

- Security of `modified` vs `unmodified` Linux system
  - system with or without SELinux
- Provides a hybrid of concepts and capabilities drawn from mandatory access controls, mandatory integrity controls, role-based access control (RBAC), and type enforcement architecture
- SELinux packages comes for several distributions
  - https://github.com/SELinuxProject/selinux#installation
  - Ubuntu, Fedora, Buildroot and of course Yocto Project

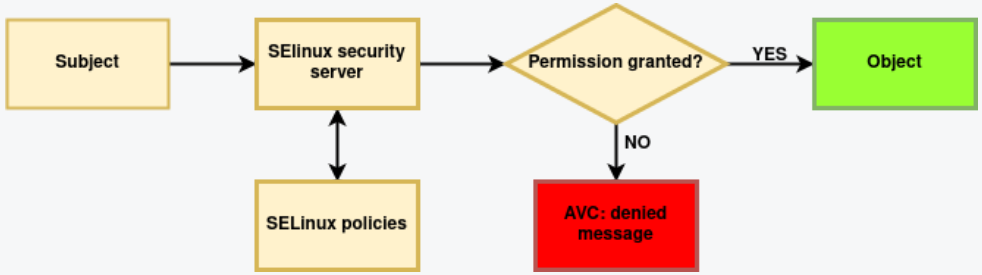
SELinux features

- Clean separation of policy from enforcement
- Well-defined policy interfaces
- Independence of specific security-label formats and contents
- Flexible policy
  - easy to write own, custom policy
- Provide controls over
  - process initialization and program execution
  - file systems, dirs, files, open file descriptors
  - sockets, network interfaces
- `Default-deny` policy
  - anything not explicitly specified in the policy is disallowed

How does it work?

- Security policies - set of rules that tell what can or can't be accessed
- All about subjects requesting access to an object
  - checking an access vector cache (AVC)
  - permission granted or denied depending on security context
- In case of denial, `avc: denied` message available in system logs

```
avc: denied { dac_override } for pid=447 comm="systemd-fstab-g" capability=1 scontext=system_u:system_r:systemd_generator_t:s0
tcontext=system_u:system_r:systemd_generator_t:s0 tclass=capability permissive=1
```



- Type enforcement and labeling
  - user:role:type:level labels format
  - policy enforcement

SELinux configuration

- There are many ways to configure SELinux to protect your system
  - most common are targeted policy and multi-level security (MLS)
  - targeted policy mostly use label type
  - MLS is way more complicated - user, role and type of label are used here
- `config` file under the `/etc`

```
# cat /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these values:
#     minimum - Minimum Security protection.
#     standard - Standard Security protection.
#     mls - Multi Level Security protection.
#     targeted - Targeted processes are protected.
#     mcs - Multi Category Security protection.
SELINUXTYPE=targeted
```

SELinux configuration

- Configuration can be read by checking on `/selinux` pseudo-file system

```
# ls /sys/fs/selinux/ -lht
total 0
dr-xr-xr-x.   2 root root    0 Nov 23 16:31 booleans
dr-xr-xr-x.   2 root root    0 Nov 23 16:31 policy_capabilities
-rw-rw-rw-.   1 root root    0 Nov 23 16:31 access
dr-xr-xr-x.   2 root root    0 Nov 23 16:31 avc
-rw-rw-rw-.   1 root root    0 Nov 23 16:31 context
--w-------.   1 root root    0 Nov 23 16:31 disable
-rw-r--r--.   1 root root    0 Nov 23 16:31 enforce
-r--r--r--.   1 root root    0 Nov 23 16:31 mls
-r--r--r--.   1 root root 2.4M Nov 23 16:31 policy
-r--r--r--.   1 root root    0 Nov 23 16:31 policyvers
```

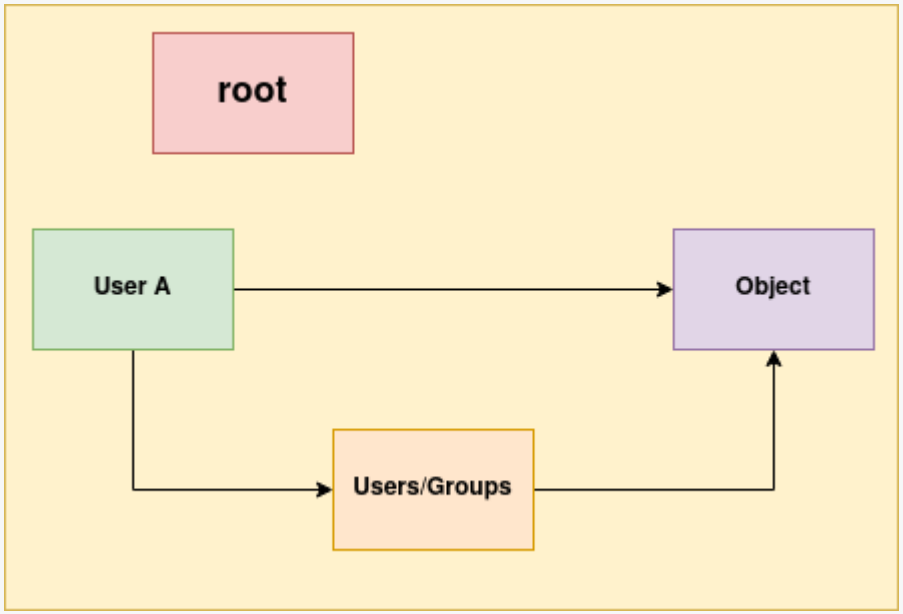
- Or by using `sestatus` command

```
# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     requested (insecure)
Max kernel policy version:      33
```

Access Control Mechanisms (ACM's)

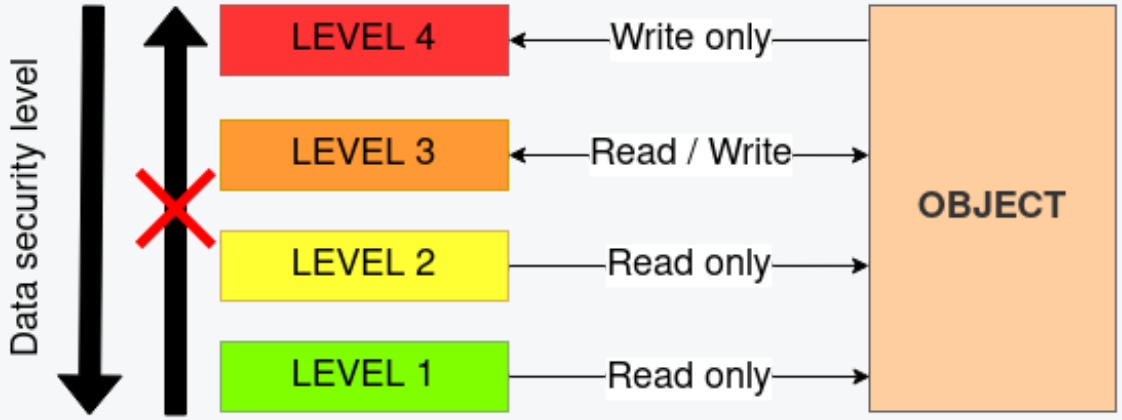
- Helps system administrators to control which users and processes can access differetn files, devices, interfaces etc.
- Examples
  - Discretionary Access Control
  - Access Control Lists
  - Mandatory Access Control
  - Role-based Access Control
  - Multi-Level Security
  - Multi-Category Security

- Discretionary Access Control
  - traditionally Linux based systems access control
  - users and groups can own a file, a process
  - ability to change permissions of own files
  - root user concept



MAC

- Mandatory Access Control
  - SELinux is one of examples
  - set of policies
  - working above DAC
  - no root user or anything similar



Policies

- Set of rules, guide the SELinux security engine
- Define `types` for file objects and `domains` for processes
- Uses `roles` to limit the `domains` that can be entered - assignes them to users
- Concept of `types` in SELinux
- Policy as a binary file

```
# ls /etc/selinux/ -lht
total 5.0K
-rw-r--r--. 1 root root  591 Mar  9  2018 config
-rw-r--r--. 1 root root 2.1K Mar  9  2018 semanage.conf
drwxr-xr-x. 4 root root 1.0K Mar  9  2018 targeted
# ls /etc/selinux/targeted/ -lht
total 4.0K
drwxr-xr-x. 4 root root 1.0K Mar  9  2018 contexts
drwx------. 2 root root 1.0K Mar  9  2018 policy
-rw-r--r--. 1 root root  539 Mar  9  2018 setrans.conf
-rw-r--r--. 1 root root   64 Mar  9  2018 seusers
```

Policies

- SELinux policies at the early stages of system start-up
  - initial SID assigning
  - mounting `/proc` - searching of `selinuxfs`
  - checking SELinux configuration
  - mounting `/selinux`
  - loading selected policy into the kernel
  - `init` re-executing, under new policy
  - start rest of boot process

```
# dmesg | grep -i selinux
[    0.238202] SELinux:  Initializing.
[    6.305588] SELinux:  policy capability network_peer_controls=1
[    6.305700] SELinux:  policy capability open_perms=1
[    6.305767] SELinux:  policy capability extended_socket_class=1
[    6.305906] SELinux:  policy capability always_check_network=0
[    6.306016] SELinux:  policy capability cgroup_seclabel=1
[    6.306127] SELinux:  policy capability nnp_nosuid_transition=1
[    6.306237] SELinux:  policy capability genfs_seclabel_symlinks=0
[    6.381543] systemd[1]: Successfully loaded SELinux policy in 343.813ms.
[    6.681054] systemd[1]: systemd 247.6+ running in system mode. (+PAM +AUDIT \
+SELINUX +IMA -APPARMOR -SMACK +SYSVINIT +UTMP -LIBCRYPTSETUP -GCRYPT -GNUTLS \
+ACL +XZ -LZ4 -ZSTD -SECCOMP +BLKID -ELFUTILS +KMOD -IDN2 -IDN -PCRE2 \
default-hierarchy=hybrid)
[    8.666298] systemd[1]: Starting SELinux autorelabel service loading...
[    8.688330] systemd[1]: Starting SELinux init for /dev service loading...
[   10.041254] systemd[1]: selinux-autorelabel.service: Succeeded.
[   10.061700] systemd[1]: Finished SELinux autorelabel service loading.
```

Policies

- `targeted policy` - example of less complex policy
- Every subject and object runs in the `unconfined_t` domain except for the specific targeted daemons
- `unconfined_t` means that there is no restrictions and the domain fall back to using DAC
- Only couple of daemons runs in their own domains - e.g. http and ntp which run in the `httpd_t` and `ntpd_t` domains, respectively

```
# ls /etc/ -Z
system_u:object_r:etc_t:s0 X11               system_u:object_r:etc_t:s0 iptables  system_u:object_r:etc_t:s0 rc5.d
system_u:object_r:alsa_etc_t:s0 asound.conf  system_u:object_r:etc_t:s0 issue     system_u:object_r:etc_t:s0 rc6.d
```

- The opposite is the `strict policy`
    - every subject and object exists in specific domain

meta-selinux overview

- Repository - https://git.yoctoproject.org/cgit/cgit.cgi/meta-selinux/
- Branches - master, dunfell, hardknott

Branch	Commit message	Author	Age
danny	Update maintainer list.	Xin Ouyang	8 years
denzil	Update maintainer list.	Xin Ouyang	8 years
dizzy	checkpolicy: remove link against libfl	Joe MacDonald	7 years
dora	psmisc: inherit enable-selinux and backport to fix build issue	Xin Ouyang	8 years
dora-next	libselinux: migrate SRC_URI to 2.2.2	Wenzong Fan	8 years
dunfell	MAINTAINERS: update email address	Armin Kuster	5 weeks
dylan	policycoreutils: fix genhomedircon construction	Joe Slater	8 years
fido	iscsi-initiator-utils: fix label for initiatorname.iscsi	Wenzong Fan	7 years
gatesgarth	libselinux-python: Fix build error due to missing target config	Anatol Belski	8 months
hardknott	secilc: Security fix for CVE-2021-36087	Armin Kuster	2 months
jethro	MAINTAINERS: Update maintainers file	Joe MacDonald	6 years
jjm/RELEASE_2.20190201	refpolicy: update to 2.20190201 and git HEAD policies	Joe MacDonald	3 years
master	coreutils/findutils/tar: remove pkgconfig from bbappend	Mingli Yu	8 weeks
master-next	libselinux.inc: Add python-shell to libselinux-python RDEPENDS.	Chris PeBenito	3 years
morty	dhcp: sync init-server with oe-core	Wenzong Fan	5 years
rocko	python-ipy: update SRC_URI to use https	Joe MacDonald	23 months
sumo	refpolicy: fix up all refpolicy 20170224 builds for sumo	Joe MacDonald	3 years
thud	refpolicy: Forward patch to apply cleanly on thud	Khem Raj	3 years
warrior	Update MAINTAINERS with new email addr	Mark Hatle	2 years
zeus	kernel: Remove non-existing kernel option	He Zhe	2 years

- Latests changes, contributions

Commits per author per quarter

Author	Q1 2021	Q2 2021	Q3 2021	Q4 2021	Total
Yi Zhao	42	0	16	0	**58**
Armin Kuster	0	2	3	0	**5**
Anatol Belski	1	0	0	0	**1**
Anibal Limon	1	0	0	0	**1**
Kai Kang	0	1	0	0	**1**
Mingli Yu	0	0	1	0	**1**
Oleksiy Obitotskyy	1	0	0	0	**1**
Philip Tricca	0	1	0	0	**1**
Total	**45**	**4**	**20**	**0**	**69**

meta-selinux README overview

- Building instructions
  - setting DISTRO_FEATURES

```
DISTRO_FEATURES_append = " acl xattr pam selinux"
```

  - specifying policy to use in build

```
PREFERRED_PROVIDER_virtual/refpolicy ?= "refpolicy-mls"
```

- By default `sysvinit` as init manager is used
- Available targets
  - `core-image-selinux`
  - `core-image-selinux-minimal`
- Additional informations
  - changing refpolicy version
  - warning of possible problems with policies

meta-selinux classes overview

- List of provided **bbclasses**

```
$ tree meta-selinux/classes/
meta-selinux/classes/
├── enable-audit.bbclass
├── enable-selinux.bbclass
├── meson-enable-selinux.bbclass
├── meson-selinux.bbclass
├── selinux.bbclass
├── selinux-image.bbclass
├── with-audit.bbclass
└── with-selinux.bbclass
```

- Most of them just enable SELinux via `PACKAGECONFIG`
- Mentions about `audit`, enabling it in `PACKAGECONFIG` as well

meta-selinux overview

- Most of the metadata are bbappends

```
$ find . -name *.bbappend
./recipes-graphics/mesa/mesa_%.bbappend
./recipes-graphics/xcb/libxcb_%.bbappend
./recipes-extended/tar/tar_%.bbappend
./recipes-extended/psmisc/psmisc_%.bbappend
./recipes-extended/pam/libpam_%.bbappend
./recipes-extended/cronie/cronie_%.bbappend
(...)
./recipes-connectivity/iproute2/iproute2_%.bbappend
./recipes-connectivity/openssh/openssh_%.bbappend
./recipes-devtools/prelink/prelink_git.bbappend
./recipes-devtools/rpm/rpm_%.bbappend
./recipes-devtools/python/python3-networkx_%.bbappend
./recipes-devtools/python/python3-decorator_%.bbappend
```

- Available policies

```
$ find . -name refpolicy*
./recipes-security/refpolicy
./recipes-security/refpolicy/refpolicy-minimum_git.bb
./recipes-security/refpolicy/refpolicy_common.inc
./recipes-security/refpolicy/refpolicy
./recipes-security/refpolicy/refpolicy-mls_git.bb
./recipes-security/refpolicy/refpolicy-targeted_git.bb
./recipes-security/refpolicy/refpolicy-standard_git.bb
./recipes-security/refpolicy/refpolicy-mcs_git.bb
./recipes-security/refpolicy/refpolicy_git.inc
```

meta-selinux overview

- Also couple of selinux specific recipes

```
$ find selinux/ -name *.bb
selinux/selinux-init_0.1.bb
selinux/libselinux_3.2.bb
selinux/libsepol_3.2.bb
selinux/mcstrans_3.2.bb
selinux/selinux-sandbox_3.2.bb
selinux/secilc_3.2.bb
selinux/libsemanage_3.2.bb
selinux/libselinux-python_3.2.bb
selinux/restorecond_3.2.bb
(...)
selinux/selinux-autorelabel_0.1.bb
selinux/checkpolicy_3.2.bb
selinux/selinux-gui_3.2.bb
selinux/selinux-dbus_3.2.bb
selinux/selinux-labeldev_0.1.bb
```

Image building

- Building core image is quite straight forward, when using README
- Build was prepared with kas tool
- On 8 cores CPU, whole build last 277 minutes

```
$ time kas-docker build meta-conference/kas.yml
(...)
Build Configuration:
BB_VERSION           = "1.50.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "universal"
TARGET_SYS           = "i686-poky-linux"
MACHINE              = "qemux86"
DISTRO               = "poky"
DISTRO_VERSION       = "3.3.4"
TUNE_FEATURES        = "m32 core2"
TARGET_FPU           = ""
                     = "<unknown>:<unknown>"
meta-oe
meta-python          = "hardknott:0b0ab6a2d227f22374268d29fcb8e4f9dab5374b"
                     = "hardknott:8b94f828a292d0e61d83aeeeeb4001c7cde08721"
meta
meta-poky
meta-yocto-bsp       = "hardknott:fbd5ce2c50c7813675185b418ab53b170416ac25"
(...)
real    276m59,506s
user    0m9,435s
sys     0m16,061s
```

Image building

- Image build for qemu machine
- `runqemu` used to start the system
    - `slirp` and `serialstdio` flags used to launch in terminal

```
$ runqemu slirp serialstdio
runqemu - INFO - Running bitbake -e ...
runqemu - INFO - Continuing with the following parameters:
KERNEL: [/work/build/tmp/deploy/images/qemux86/bzImage--5.10.76+git0+e1979ceb17_be6faea8fd-r0-qemux86-20211122162655.bin]
MACHINE: [qemux86]
FSTYPE: [ext4]
ROOTFS: [/work/build/tmp/deploy/images/qemux86/core-image-selinux-qemux86-20211123170013.rootfs.ext4]
CONFFILE: [/work/build/tmp/deploy/images/qemux86/core-image-selinux-qemux86-20211123170013.qemuboot.conf]
runqemu - INFO - Network configuration: ip=dhcp
runqemu - INFO - Port forward: hostfwd=tcp::2222-:22 hostfwd=tcp::2323-:23
runqemu - INFO - Running /work/build/tmp/work/x86_64-linux/qemu-helper-native/1.0-r1/recipe-sysroot-native/usr/bin/qemu-system-i386 \
-device virtio-net-pci,netdev=net0,mac=52:54:00:12:35:02 -netdev user,id=net0,
hostfwd=tcp::2222-:22,hostfwd=tcp::2323-:23,tftp=/work/build/tmp/deploy/images/qemux86 \
-object rng-random,filename=/dev/urandom,id=rng0 -device virtio-rng-pci,rng=rng0 \
-drive file=/work/build/tmp/deploy/images/qemux86/core-image-selinux-qemux86-20211123170013.rootfs.ext4,if=virtio,format=raw \
-usb -device usb-tablet   -cpu core2duo -m 256 -serial mon:stdio -serial null  \
-kernel /work/build/tmp/deploy/images/qemux86/bzImage--5.10.76+git0+e1979ceb17_be6faea8fd-r0-qemux86-20211122162655.bin \
-append 'root=/dev/vda rw  mem=256M ip=dhcp console=ttyS0 console=ttyS1 oprofile.timer=1`
[    0.000000] Linux version 5.10.76-yocto-standard (oe-user@oe-host) (i686-poky-linux-gcc (GCC) 10.2.0, \
GNU ld (GNU Binutils) 2.36.1.20210209) #1 SMP PREEMPT Fri Oct 29 01:33:22 UTC 2021
```

Starting QEMU image

- Checking status of SELinux in booted system

```
root@qemux86:~# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     requested (insecure)
Max kernel policy version:      33
```

- Checking system logs on boot

```
# dmesg | grep -i selinux
[    0.249275] SELinux:  Initializing.
[    5.511707] SELinux:  policy capability network_peer_controls=1
[    5.511825] SELinux:  policy capability open_perms=1
[    5.511981] SELinux:  policy capability extended_socket_class=1
[    5.512117] SELinux:  policy capability always_check_network=0
[    5.512263] SELinux:  policy capability cgroup_seclabel=1
[    5.512468] SELinux:  policy capability nnp_nosuid_transition=1
[    5.512612] SELinux:  policy capability genfs_seclabel_symlinks=0
[    5.584558] systemd[1]: Successfully loaded SELinux policy in 223.105ms.
[    5.778565] systemd[1]: systemd 247.6+ running in system mode. (+PAM +AUDIT +SELINUX \
+IMA -APPARMOR -SMACK +SYSVINIT +UTMP -LIBCRYPTSETUP -GCRYPT -GNUTLS +ACL +XZ -LZ4 -ZSTD \
-SECCOMP +BLKID -ELFUTILS +KMOD -IDN2 -IDN -PCRE2 default-hierarchy=hybrid)
[    7.663127] systemd[1]: Starting SELinux autorelabel service loading...
[    7.708807] systemd[1]: Starting SELinux init for /dev service loading...
[    9.001358] systemd[1]: selinux-autorelabel.service: Succeeded.
[    9.028820] systemd[1]: Finished SELinux autorelabel service loading.
```

- Checking system logs for audit inputs

```
Nov 23 17:54:50 qemux86 audit[215]: AVC avc:  denied  { bpf } for  pid=215 comm="systemd" capability=39  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
Nov 23 17:54:50 qemux86 audit[215]: AVC avc:  denied  { bpf } for  pid=215 comm="systemd" capability=39  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
Nov 23 17:54:50 qemux86 audit[215]: AVC avc:  denied  { bpf } for  pid=215 comm="systemd" capability=39  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
Nov 23 17:54:50 qemux86 audit[215]: AVC avc:  denied  { perfmon } for  pid=215 comm="systemd" capability=38  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
Nov 23 17:54:50 qemux86 audit[215]: AVC avc:  denied  { perfmon } for  pid=215 comm="systemd" capability=38  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
Nov 23 17:54:50 qemux86 audit[215]: AVC avc:  denied  { perfmon } for  pid=215 comm="systemd" capability=38  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
Nov 23 17:54:51 qemux86 systemd[1]: Started Session c1 of user root.
Nov 23 17:54:51 qemux86 audit[204]: USER_LOGIN pid=204 uid=0 auid=4294967295 ses=4294967295 \
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023 msg='op=login acct="root" exe="/bin/login.shadow" \
hostname=qemux86 addr=? terminal=/dev/ttyS0 res=success'
```

QEMU image - working with access denial

- Using `audit2allow`

```
root@qemux86:~# audit2allow --help
Usage: audit2allow [options]

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -b, --boot            audit messages since last boot conflicts with -i
  -a, --all             read input from audit log - conflicts with -i
  -p POLICY, --policy=POLICY
                        Policy file to use for analysis
  -d, --dmesg           read input from dmesg - conflicts with --all and
                        --input
  -i INPUT, --input=INPUT
                        read input from <input> - conflicts with -a
  -l, --lastreload      read input only after the last reload
  -r, --requires        generate require statements for rules
  -m MODULE, --module=MODULE
                        set the module name - implies --requires
  -M MODULE_PACKAGE, --module-package=MODULE_PACKAGE
                        generate a module package - conflicts with -o and -m
  -o OUTPUT, --output=OUTPUT
                        append output to <filename>, conflicts with -M
  -D, --dontaudit       generate policy with dontaudit rules
  -R, --reference       generate refpolicy style output
  -N, --noreference     do not generate refpolicy style output
  -v, --verbose         explain generated output
  -e, --explain         fully explain generated output
  -t TYPE, --type=TYPE  only process messages with a type that matches this
                        regex
  --perm-map=PERM_MAP   file name of perm map
  --interface-info=INTERFACE_INFO
                        file name of interface information
  -x, --xperms          generate extended permission rules
  --debug               leave generated modules for -M
  -w, --why             Translates SELinux audit messages into a description
                        of why the access was denied
```

QEMU image - working with access denial

- Best to use `audit2allow -a -w`

```
# audit2allow -a -w
type=AVC msg=audit(1637690013.758:51): avc:  denied  { map } for  pid=183 comm="dbus-daemon" \
path="/sys/fs/selinux/status" dev="selinuxfs" ino=19 scontext=system_u:system_r:system_dbusd_t:s0-s0:c0.c1023 \
tcontext=system_u:object_r:security_t:s0 tclass=file permissive=0
        Was caused by:
                Missing type enforcement (TE) allow rule.

                You can use audit2allow to generate a loadable module to allow this access.

type=AVC msg=audit(1637690090.567:74): avc:  denied  { bpf } for  pid=215 comm="systemd" capability=39  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
        Was caused by:
                Missing type enforcement (TE) allow rule.

                You can use audit2allow to generate a loadable module to allow this access.

type=AVC msg=audit(1637690090.567:74): avc:  denied  { perfmon } for  pid=215 comm="systemd" capability=38  \
scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 \
tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=capability2 permissive=0
        Was caused by:
                Missing type enforcement (TE) allow rule.

                You can use audit2allow to generate a loadable module to allow this access.
(...)
```

Summary

- SELinux provide very configurable access control system
- The entry level seems high, but fortunately the basic security features are easy to implement
- meta-selinux is easy to integrate into a build, but from experience I can say that it's best to add it at an early stage of system design
- Discussed layer provide set of tools that help with implementing access control

Contact us

We are open to cooperate and discuss

- contact@3mdeb.com
- facebook.com/3mdeb
- @3mdeb_com
- linkedin.com/company/3mdeb
- https://3mdeb.com
- Book a call
- Sign up for the newsletter

Feel free to contact us if you believe we can help you in any way. We are always open to cooperate and discuss.

3MDEB

Q&A